Accelerable Lottery Tickets with the Mixed-Precision Quantization

Zhangheng Li
UT Austin
zoharli@utexas.edu

Yu Gong
Rutgers University
yg430@soe.rutgers.edu

Zhenyu Zhang
UT Austin
zhenyu.zhang@utexas.edu

Xingyun Xue
UT Austin
xuexingyun@utexas.edu

Tianlong Chen
UT Austin
tianlong.chen@utexas.edu

Yi Liang
Google Research
yiliang@google.com

Bo Yuan
Rutgers University
bo.yuan@soe.rutgers.edu

Zhangyang Wang
UT Austin
atlaswang@utexas.edu

Abstract

In recent years, the lottery tickets hypothesis has gained widespread popularity as a means of network compression. However, the practical application of lottery tickets for hardware acceleration is difficult due to their element-wise unstructured sparsity nature. In this paper, we argue that network pruning can be seen as a special case of network quantization, and relax the hard network pruning with mixed-precision quantization in an unstructured manner, which makes it possible for real hardware acceleration. We successfully validate the wide existence of quantized lottery tickets, namely MPQ-tickets, that can match or even surpass the performance of corresponding full-precision dense networks on various representative benchmarks. Also, we demonstrate that MPQ-tickets have much higher flexibility than vanilla lottery tickets, and largely benefit from pruning when compared to QNNs. Moreover, the MPQ-tickets achieve up to $8\times$ *hardware acceleration of inference speed and* $14\times$ *less memory consumption than full-precision models.*

1. Introduction

In recent years, neural network efficiency gradually becomes an important topic due to growing model size, especially for large-scale pre-training models, and researchers seek for effective ways to compress neural networks while preserving model performance. Among different neural network compression methods, quantization and pruning are two popular choices, both are easy to implement and achieve good trade-off in performance and compression. However, few works have investigated unifying quantizaiton and pruning, one of which [12] firstly proposed that pruning can be treated as 0-bit quantization. While providing such a novel perspective, they didn't further explore how to combine the advantages of quantization and pruning. As we know, the main advantage of quantization is hardware acceleration, however, the performance might drop when networks are quantized to low bitwidth, such as 1-bit or 2-bit. Meanwhile, some competitive unstructured pruning methods, such as Iterative Magnitude Pruning (IMP) in Lottery Ticket Hypothesis(LTH) [6], can better preserve or even surpass the performance of original dense models, even with very high sparsity. However, unstructured pruning is known to have an inability for real-hardware acceleration.

Based on this point, we take a step further and argue that quantization can be seen as soft pruning compared to hard pruning that aggressively turns 32-bit weight to 0-bit, and by gradually decreasing the quantizing precisions of the weights, the pruning process enjoys a smooth transition, where eventually parts of the weights are pruned completely and parts of the weight remains low-precision quantization. Note that this is different from prior works on mixed-precision quantization [2, 9, 13, 15–18] where the bitwidth of the weights is decided either by learning or searching, instead, we monotonically decrease the bitwidth of weights towards being completely pruned, which is why we call this process as soft pruning. Besides, to utilize the advantage of unstructured pruning which prune each weight element separately, we first explore element-wise mixed-precision quantization, which is also different from previous works that only considered layer-wise mixed-precision. The output networks produced by soft pruning can enjoy customized hardware acceleration of mixed-precision quantization, and also has the advantage of unstructured pruning that can match or surpass the performance of original dense networks.

Subsequently, a core problem is how to gradually decrease the bitwidth of the weights such that we can find mixed-precision quantized subnetworks that have good performance and compression trade-off. In this paper, we introduce the softened version of IMP from LTH [6], namely the Iterative Magnitude Quantization (IMQ), by iteratively choosing a certain portion of weights with the least full-precision magnitude and halving their bitwidth. Using IMQ, we empirically demonstrate that winning lottery tickets can also be found, namely the Mixed-Precision Quantization Tickets (MPQ-tickets), that can surpass the performance of original lottery tickets and other competitive baselines and realize 5.79-8.63 hardware acceleration compared to full-precision models and original lottery tickets that are hard to be accelerated.

Specifically, our contributions are as follows:

- We propose a novel kind of lottery ticket named *MPQ-tickets*, by relaxing the hard pruning in vanilla lottery tickets with soft pruning, i.e. iterative quantization. We demonstrate the wide existence of MPQ-tickets under various settings.
- We compare MPQ-tickets with several strong baselines, including vanilla lottery tickets and QNNs, and demonstrate that MPQ-tickets outperform these baselines in terms of performance and compression trade-off. We analyze and empirically validate that MPQ-tickets have much higher flexibility than vanilla lottery tickets, and largely benefit from pruning when compared to QNNs.
- We implement hardware support for MPQ-tickets, by building the global buffer to store the operands element-wise quantized and configure a single DSP slices on-chip to be a multiple channel multiplier for processing multiple data simultaneously. We demonstrate that the inference speed and memory consumption can be reduced by 8.67$\times$ and 14.57$\times$ with accuracy performance comparable to full-precision models.

2. Related Work

2.1. Network Quantization

Network quantization compresses the original network by reducing the number of bits required to represent each weight. DoReFa-Net [19] is a representative quantization method with bit convolution kernels and STE gradient back-propagations. As previous works typically clip the activations to a constant interval and then perform quantization, [3] proposed PACT for activation quantization that learns adaptive clip thresholds which better preserves the network expressiveness. DoReFa-Net and PACT are adopted as weight-quantization and activation-quantization methods in our work, respectively, but note that our proposed method is quantization method-agnostic.

2.2. Joint Pruning and Quantization

Several works have explored combining the two effective model compression techniques of pruning and quantization. [10] proposed parallel pruning-quantizatin with full precision fine-tuning, [14] further proposed a differentiable loss to jointly optimize pruning and quantization. However, these two works didn't exploit connections between pruning and quantization. [12] firstly provided the viewpoint that pruning can be seen as 0-bit quantization, and decided weight precisions by learning. Notably, this work produces the final quantizing precisions by learning, while our work decreases quantizing precisions monotonically towards 0-bit quantization, i.e. hard pruning.

2.3. Mixed Precision Quantization

To better leverage the trade-off between performance and compression of QNNs, lots of prior works explored mixed-precision quantization techniques which can improve this trade-off. Many of these works mainly focus on the strategies to find best quantizing precision for each layer, such as Neural Architecture Search (NAS)-based [16, 18], Attribution Rank Preservation-based [15], constrained optimization-based [2, 9], bit-level sparsity regularizer-based [17], and hardware feedback-awared searching-based [13] methods. However, previous works only considered layer-wise mixed quantization, i.e. the quantizing precision is different among different layers but the same inside each layer, whereas our work for the first time investigates element-wise mixed precision, i.e. each weight element has its own quantizing precision.

2.4. Quantized Neural Network Accelerator

Various platforms are adopted to implement the quantized neural network, including GPU, FPGA, and ASIC. Current GPU supports the operand quantized with 1-bit, 4-bit, 8-bit and 16-bit [4]. However, the limited precision prohibits the performance potential for the mixed precision network. FPGA has rich DSP and LUT resources to support the mixed precision network, due to the reconfigurability. [11] utilizes the LUT resource to build bit-serial computation units, supporting various precision from layer to layer. [7] and [1] have the ability to compute operands with different precision intra a layer. On ASCI design, [8] and [5] reduce statically ineffectual bits from activation data to improve the latency.

3. Methodology

3.1. Preliminaries and Notations

Lottery Ticket Hypothesis(LTH) For a network $f(x;\theta)$ with input samples x and model parameters θ, a sparse sub-network is a network $f(x;\theta \odot m_b)$ with a binary mask $m_b \in \{0,1\}^{|\theta|}$, where $\odot$ is the element-wise product. In

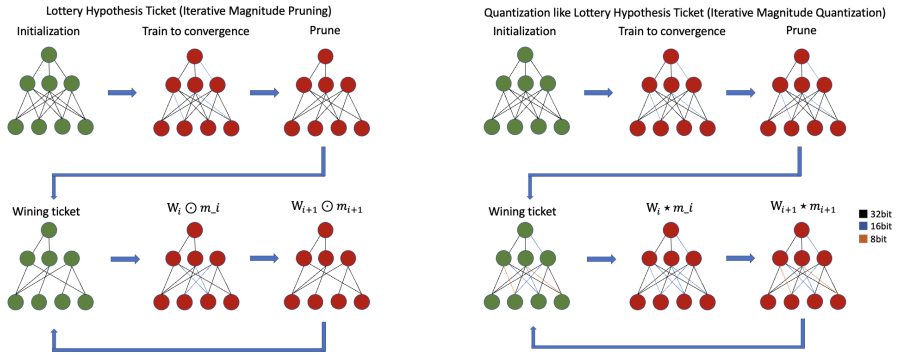


Figure 1. The flow diagrams and comparison of IMP and our proposed IMQ. The colored lines are weight connections with globally least magnitude that are chosen to be pruned or quantized. (Left) the IMP flow: the network is iteratively trained and hard-pruned using least weight magnitude criterion with weight-rewinding. (Right) the IMQ flow: the network is iteratively trained and quantized with decreasing bitwidth using least weight magnitude criterion with weight-rewinding. Note that the weight magnitudes are ranked in full-precision at each iteration and we preserve the quantization results with masks. Also note that although we demonstrate the example using MLP, in experiments we only quantize or prune weights of convolutional layers.

order to find sparse sub-networks that can match the performance of their dense counterparts, [6] proposed the **Iterative Magnitude Pruning**(IMP) algorithm : Start from a dense initialization W_0, train the network until convergence to weight W_t. Then we determine the ρ percent smallest magnitude weights in $|W_t|$ and create a binary mask m_0 that prunes these. Then retrain the pruned network from the same initialization weight $W_0 \odot m_0$ to convergence. Iterating this procedure will produce subnetworks with different sparsity which can usually match the performance of original dense networks, a.k.a. the winning tickets.

DoReFa-Net The core quantization function and corresponding backward Straight-Through Estimator (STE) in DoReFa-Net are defined as follows:

$$\textbf{Forward}: r_0 = \frac{1}{2^k - 1} round((2^k - 1) r_i) \tag{1}$$

$$\textbf{Backward}: \frac{\partial c}{\partial r_i} = \frac{\partial c}{\partial r_o} \tag{2}$$

where $r_i \in [0, 1]$ is a input real number, $r_o \in [0, 1]$ is the quantized k-bit number output and c is the objective loss function. Following [19], we call this operation $quantize_k$ where k indicates the bitwidth after quantization. Then the weights in DoReFa-Net are quantized as follows:

$$\textbf{Forward}: r_0 = 2 \times quantize_k\left(\frac{\tanh(r_i)}{2\max(|\tanh(r_i)|)} + \frac{1}{2}\right) - 1 \tag{3}$$

$$\textbf{Backward}: \frac{\partial c}{\partial r_i} = \frac{\partial r_o}{\partial r_i}\frac{\partial c}{\partial r_o} \tag{4}$$

where r_i is the original full-precision weight and r_o is the quantized weight. With equation 3, the weights are quantized to [-1,1].

PACT PACT is an effective activation quantization scheme in which the activation function has a learnable clipping threshold. The PACT activation is defined as follows:

$$y = PACT(x) = 0.5(|x| - |x - \alpha| + \alpha) = \begin{cases} 0, & x \in (-\infty, 0) \\ x, & x \in [0, \alpha) \\ \alpha, & x \in [\alpha, +\infty) \end{cases} \tag{5}$$

where α is a learnable scalar for each layer of the neural network. The clipped activation is then quantized using the method similar to the $quantize_k$ function in DoReFa-Net:

$$y_q = round\left(y \cdot \frac{2^k - 1}{\alpha}\right) \cdot \frac{\alpha}{2^k - 1} \tag{6}$$

Where y_q is the quantized activation. STE is adpoted for gradient-based training of α. See [3] for more details.

3.2. MPQ-tickets: The Mixed-precision Lottery Tickets

LTH has proved its wide existence through rich experiments. However, it has two drawbacks: (1) Subnetworks obtained by unstructured pruning cannot be accelerated on hard-ware level. (2) We argue that the aggresive hard pruning in LTH will hurt the final performance.

To address these 2 problems, we propose MPQ-ticket—— by relaxing the hard binary mask in original lottery tickets to quantized mask of different bitwidth, and use the mask to quantize weight by extending layerwise quantization technique in DoreFa-Net to element-wise. The overall flow diagram is shown in Figure 1. For intuitive understanding, consider pruning as a special case of quantization from 32-bit to 0-bit. To mitigate such big bitwidth gap, we extend the original two stage (32-bit and 0-bit) to fine-grained

hierarchy(32-bit, 16-bit, 8-bit, 4-bit, 2-bit, 1-bit, 0-bit). However, we find that involving 2-bit and 1-bit quantization can significantly reduce the network performance, which phenomenon is in consistency with previous empirical results of QNNs [3, 19] showing dropped performance at 1-bit or 2-bit quantizations. We thus remove 2-bit and 1-bit, and our resulting quantization hierarchy is (32-bit, 16-bit, 8-bit, 4-bit, 0-bit).

The process of finding MPQ-tickets resembles that of LTH such as the usage of iterative magnitude pruning (IMP) and weight rewinding during iterative training, but except for the obtained mask and the pruning operation with the mask which serves our quantization purpose. We call the adapted version of IMP as Iterative Magnitude Quanztization (IMQ), and next introduce IMQ, weight rewinding and quantization mask in our method respectively:

Iterative Magnitude Quantization At each iteration, the network is firstly trained to convergence, and then we choose a portion of weights with least magnitude and not equal to 0, and update the bidwidth mask obtained from last iteration to halve the bitwidth of these weights. For example, if a weight is quantized only once, then it is quantized from 32-bit float-point to 16-bit fixed-point, and the corresponding bitwidth mask value will be 16; if a weight is quantized for 3 times iteratively, then its bitwidth mask value will be 4.

Weight Rewinding Before the training of each iteration, we firstly rewind the network weights to the initial weights of 1st iteration, then perform quantization on the initial weights according to the bidwidth mask obtained from last iteration and start training.

Quantization Mask At 1st iteration, we initialize the mask with full precision: $m_q = \{32\}^{|\theta|}$, and gradually update it to $m_q \in \{0, 4, 8, 16, 32\}^{|\theta|}$ through IMQ which indicates the current bitwidth of each weight. The network after quantization can be denoted as $f(x; \theta \star m_q)$ where $\star$ is the quantization operation. In this paper, we adapt the weight quantization operation in DoReFa-Net and extend its layer-wise quantization to element-wise quantization. Specifically, for k-bit($k > 1$) weight, we use $quantize_k$ as mentioned above as the quantization function; for 0-bit weight, the result is similar to one in IMP. The above quantization function Q can be formally presented as follows:

$$Q(w_i, m_i) = \begin{cases} 0 & m_i = 0 \\ quantize_{m_i}(w_i) & m_i \in 4, 8, 16, 32 \end{cases} \quad (7)$$

where w_i is the 32-bit float-point weight element, m_i is the corresponding bitwidth mask element. For activation quantization, we use PACT [3] which is able to match the performance of full-precision models when quantized to no less than 4-bit. Note that different from element-wise weight quantization, we uniformly quantize activation to a fixed bitwidth.

The main performance advantage of MPQ-tickets is twofold: firstly, compared to vanilla lottery tickets, MPQ-tickets relax the hard pruning with soft pruning such that it introduces exponentially more soft pruning states, among which lottery tickets with better performance can be found; secondly, compared to vanilla QNNs and other mixed-precision networks that don't necessarily guarantee the monotonic decrease of average bitwidth from prior works, MPQ-tickets largely benefits from pruning that monotonically remove redundant weights. Besides, MPQ-tickets have the compression advantage over vanilla lottery tickets in that they can achieve real hardware acceleration with our customized hardware support.

We elaborate the IMQ algorithm of finding MPQ-tickets in the Appendix.

3.3. FPGA Based Accelerator Design

Due to the reconfigurability, we choose FPGA as the platform to implement various neural network with element-wise quantization. Basic DSP slices on FPGA can process the digital multiplication and accumulation in a high speed to

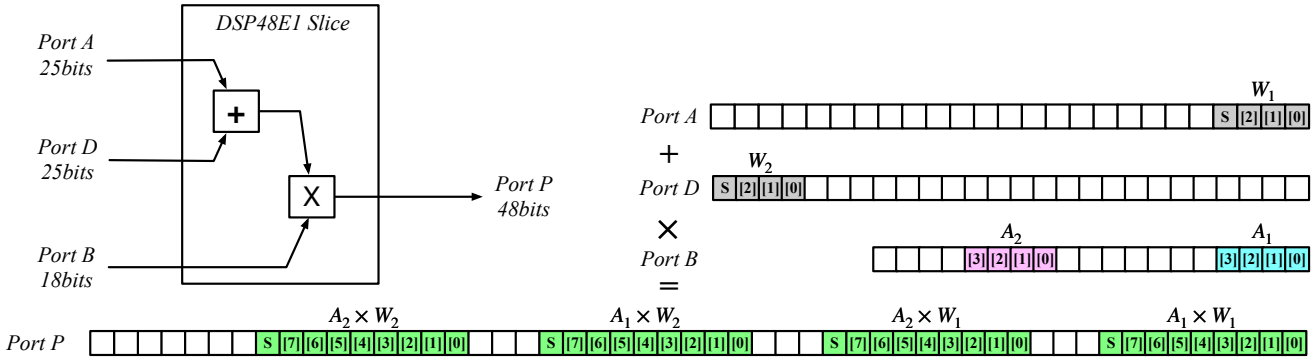
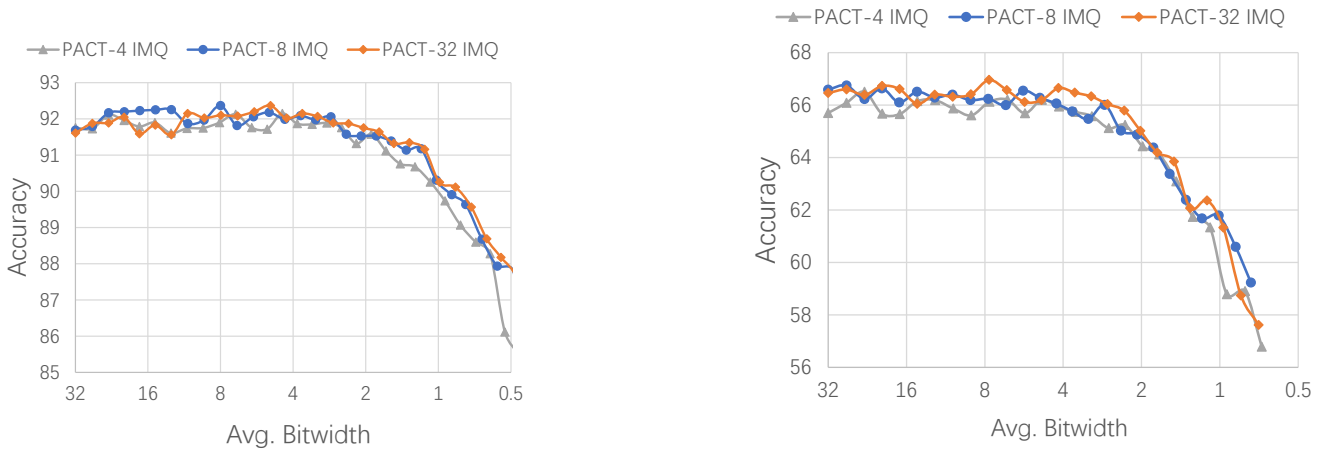


Figure 2. Four channels for 4-bit multiplication in a DSP slice.



(a) CIFAR10 results

(b) CIFAR100 results

Figure 3. The existence of MPQ-tickets at varying settings. PACT-k denotes the PACT activation are quantized to k-bit. (a) IMQ results on CIFAR10 dataset when PACT are quantized to 4,8,32 bits. (b) IMQ results on CIFAR100 dataset when PACT are quantized to 4,8,32 bits.

Algorithm 1 IMQ algorithm for find MPQ-tickets

t

Input: Model $f(x;\theta_0)$, $m_q = \{32\}^{|\theta|}$, iteration N_{iter}, quantize ratio ρ.
Output: mask $m_q \in \{0,4,8,16,32\}^{|\theta|}$, MPQ-tickets $f(x;\theta^* \star m_q)$.
$D_{best} \leftarrow D_1$
for $k = 0$ **to** $N_{iter} - 1$ **do**
    Rewind network parameters θ to θ_0
    Quantize network with current m_q: $f = f(x;\theta_0 \star m_q)$
    Train $f = f(x;\theta_0 \star m_q)$ to converge, get network parameter
    Find least magnitude parameter subset θ' where $|\theta'| = \rho|\theta|$.
    Update m_q by halving the corresponding value w.r.t. θ'.
    $m_q^k = m_q$
end for
Get MPQ-tickets from $f(x;\theta_k^* \star m_q^k)$ where the test set accuracy is bigger than $f(x;\theta_0^* \star m_q^0)$
return D_{best}

enhance the speed of inference. To exploit the ability of DSP, when the operands are quantized with low bits (8-bit, 4-bit, 2-bit and 1-bit), multi low-bit data are combined to a single high-bit data and sent to the DSP slices in a cycle. Taking the basic *DSP48E1* slice as an example, it integrates a pre-adder and a 25×18 multiplier. As shown in Figure 2, a DSP slice is configured to a four-channel multiplier for 4-bit operands. In similar way, a DSP slice can configured to process multi low-bit operands simultaneously, with the mask to denote the operand bit width. The element-wise quantized neural network reduces data size to address the bottleneck of data access. However, the buffers on FPGA are not well-utilized, which increase the data access and waste the resource on chip. We build the global buffer of 32-bit width to store the mixed precision data, with the mask indicating the bit-with of each data. The computation flow on the accelerator is as follows: first, the quantized the data is fetched from DDR off chip to store the global buffers on chip, with the mask denoting the bit width, then, multi data are combined and processed in one cycle to accelerate the computation, at last, the result stored in the buffer is sent to the DDR again.

4. Experiments

In this section, we validate the existence of our proposed MPQ-tickets and evaluate the performance and compression trade-off with experiments. Specifically, we try to answer 3 major questions: (1). Do MPQ-tickets widely exist just like vanilla lottery tickets? (2) How can MPQ-tickets outperform vanilla lottery tickets, vanilla QNNs, and other baselines in terms of performance and compression trade-off? (3) How does our customized hardware support accelerate the inference speed of MPQ-tickets?

Experiment setup. We benchmark all experiments using ResNet20s on two classical datasets, CIFAR10 and CIFAR100. We train the networks using IMP, IMQ and vanilla quantizations (denoted as QNN or QNN-k for k-bit weight quantization), and compare their performance. For sanity check, we also compare Iterative Random Pruning and Iterative Random Quantization by replacing the least weight magnitude criterion with random choosing, and we denote them as IRP and IRQ, respectively. For IMQ and IRQ that

use PACT for activation quantization, we use *PACT-k* prefix to denote the activations are quantized to *k*-bit, and mainly explore the setting when *k*=32,8,4. For IMP and IRP, we use 32-bit full precision and set the pruning rate at each iteration as 0.2, and repeat for 16 iterations. For IMQ and IRP, we set the quantization rate as 0.3, and repeat for 25 iterations. For evaluation criterion, we mainly evaluate the performance and compression trade-off using test accuracy v.s. average bitwidth trade-off. For hardware acceleration evaluation, we simply evaluate the inference speed of our compressed models.

Next, we present our experiment results that answers the 3 questions we propose in the start of this section.

Table 1. The performance comparison of different instances of ResNet20s on CIFAR-10 dataset. The *PACT* model performance is from the orignal paper [3]. For LTH and random MPQ-tickets, we simply report the best results.(PACT-k, q-bit) denotes the best performing lottery ticket with average weight bitwidth below q-bit and activation quantized to k-bit using PACT.

Model	Accuracy
Full Precision	91.81
LTH	92.04
PACT(5-bit) [3]	91.7
PACT(4-bit) [3]	91.3
QNN (PACT-32, 8-bit)	91.75
QNN (PACT-32, 4-bit)	91.58
QNN-ticket (PACT-32, 4-bit)	91.87
QNN-ticket (PACT-32, 2-bit)	91.65
Random MPQ-ticket	91.84
MPQ-ticket (PACT-32, 8-bit)	**92.37**
MPQ-ticket (PACT-32, 4-bit)	92.15
MPQ-ticket (PACT-32, 2-bit)	91.64
MPQ-ticket (PACT-8, 8-bit)	**92.37**
MPQ-ticket (PACT-8, 4-bit)	92.09
MPQ-ticket (PACT-8, 2-bit)	91.53
MPQ-ticket (PACT-4, 8-bit)	92.15
MPQ-ticket (PACT-4, 4-bit)	91.89
MPQ-ticket (PACT-4, 2-bit)	91.58

4.1. The Existence of MPQ-Tickets

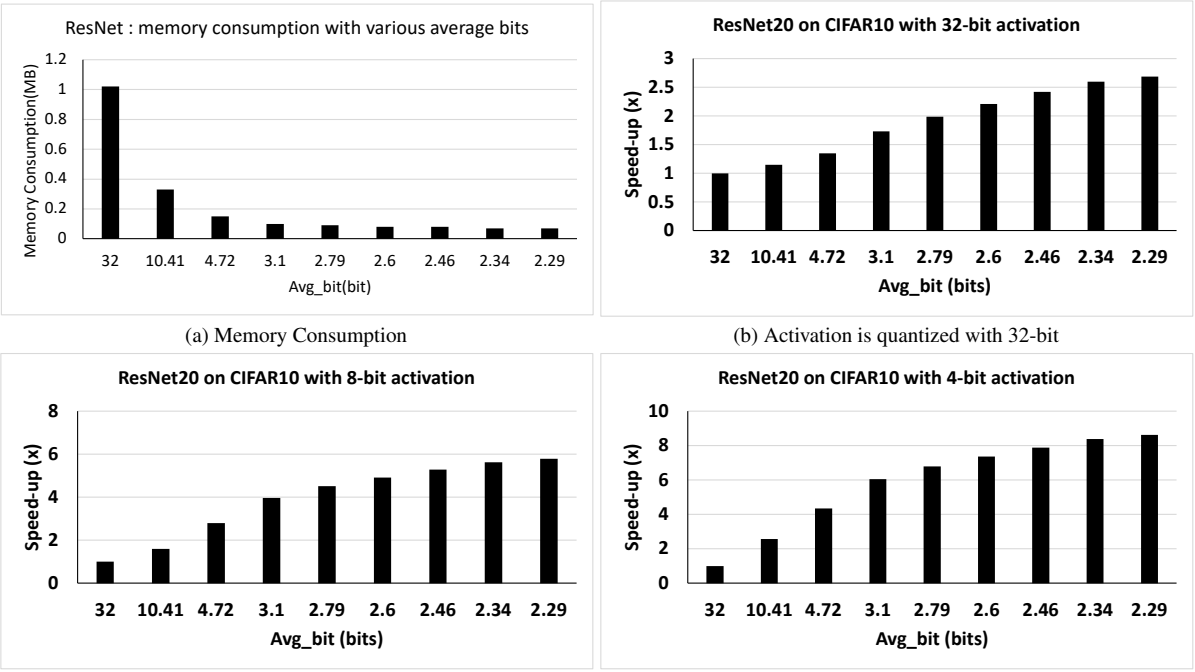
We first validate the existence of MPQ-tickets. To achieve this, we train ResNet20s with IMQ and varying PACT-k (which would influence hardware acceleration) on CIFAR10 and CIFAR100 datasets, and the test results are shown in Figure 3. We observe that under various setting, the subnetworks produced by IMQ can match the performance of 32-bit dense models when average bitwidth is greater than around 4 bit, and then start to decrease when average bitwidth continues to decrease. We therefore successfully validate the wide existence of MPQ-tickets. We also note that the subnetwork accuracies produced by PACT-8 IMQ is comparable to those by PACT-32 IMQ on both dataset, which is favorable because PACT-8 subnetworks have significantly faster inference speed than PACT-32 subnetworks at the same average weight bitwidth, as we will show in the following subsection. The subnetwork accuracies produced by PACT-4 IMQ are slightly lower, but their inference speed would be further improved which provide trade-off choices in practice.

4.2. How does the performance and compression trade-off of MPQ-tickets outperform other baselines?

To evaluate the performance and compression trade-off, we compare MPQ-tickets produced by PACT-8 IMQ with various baselines, namely the PACT-8 IRQ, IMP, IRP, QNN, QNN-8 IMP, and the testing curves are shown in Figure 8. Overall speaking, our proposed MPQ-tickets found by IMQ consistently outperform other baselines when average bitwidth is greater than 3-bit, despite the initial performance is worse than some baselines.

Comparison with vanilla lottery tickets By comparing the curves of IMP and PACT-8 IMQ from Figure 8, we validate the effectiveness of MPQ-tickets compared to vanilla lottery tickets. As we discussed in Section 1 and 3, the soft pruning manner of MPQ-tickets is more flexible than hard pruning of vanilla lottery tickets in that soft pruning can find exponentially more intermediate pruning states than hard pruning, and thus has higher possibility to find lottery tickets with better performance. What's more, due to the unstructured pruning nature of vanilla lottery tickets, they are technically hard to be accelerated on hardware, whereas MPQ-tickets are able to be significantly accelerated as we will show in next subsection. We therefore conclude the advantages of MPQ-tickets compared to vanilla lottery tickets.

Comparison with vanilla QNNs We compare vanilla QNNs with the same weight and activation quantization methods (i.e. DoReFat-Net and PACT) as adpoted in MPQ-tickets. The differences are that vanilla QNNs quantize weights uniformly to a fixed precision and are trained only once, while MPQ-tickets quantize weights with element-wise precision and are trained iteratively. We keep all other hyperparameters the same, and by comparing the testing curves of QNN and PACT-8 IMQ in Figure 8, we observe that MPQ-tickets consistently outperform vanilla QNNs at the same average bitwidth. We therefore validates the effectiveness of MPQ-tickets over vanilla QNNs. The reasons of this effectiveness are twofold: firstly, MPQ-tickets enjoy the advantages of pruning which can remove redundant weight connections; secondly, due to mixed-precision nature, MPQ-tickets are more flexible than vanilla QNNs in allocating bitwidth than vanilla QNNs.



(a) Memory Consumption

(b) Activation is quantized with 32-bit

(c) Activation is quantized with 8-bit

(d) Activation is quantized with 4-bit

Figure 4. The FPGA platform memory consumption and speed-up of ResNet20s with various average bit-width. (a) The memory consumption ResNet20s on FPGA with various average bit. (b) when the activation is quantized with 32-bit, the optimal speed-up is 2.69×. (c) when the activation is quantized with 8-bit, the optimal speed-up is 5.79×. (d) when the activation is quantized with 4-bit, the optimal speed-up is 8.63×.

Other Comparisons One might note that since the performance of vanilla QNNs at 8-bit or 4-bit quantization is comparable to full-precision counterparts, it's hopeful to perform IMP over QNNs to obtain lottery tickets of QNNs that have much lower average bitwidth. Consequently, how do IMP over QNNs compare with MPQ-tickets? We perform an experiment of IMP over an 8-bit QNN, denoted as *QNN-8 IMP*. As we see from the results of Figure 8, MPQ-tickets can still outperform QNN-8 IMP at around 3-8 average bitwidth, we believe this is due to the mixed-precision nature of MPQ-tickets that can flexibly allocate more bitwidth to important weights and less to redundant weights.

We notice that the initial performance of IMQ at 1st iteration is lower than the full-precision counterpart of IMP at 1st iteration. We conjecture this is because the activation is clipped in PACT, and thus limiting the expressiveness of the network. To validate it, we conduct an experiment to replace ReLU activation with PACT, but remove the quantization part of PACT (i.e. only remain the parameterized clipping), and then run full-precision IMP training. We denote this experiment as *PACT-full IMP*, and compare the results in Figure 5. We observe significant performance degradation of PACT-full IMP compared to IMP, which validates our conjecture. However, under such circumstance, MPQ-tickets still outperform other baselines. We believe that with better activation quantization techniques, the performance of MPQ-tickets can be further boosted, which is orthogonal to our contributions.

As we mentioned in Section 3, we empirically find that IMQ with 1-bit and 2-bit included in quantization hierarchy performs badly compared to 1-bit and 2-bit excluded, as shown in Figure 6. This is in fact a common phenomenon in prior QNN works, where the performance starts to degrade when the bitwidth is lower than 4-bit. We believe this is because the round function at 1-bit and 2-bit precision make the full-precision weights change too significantly that hurt the network performance.

Finally, we also present the best results from different models, as shown in Table 1. We observe that our implemented QNNs match the performance of the original PACT paper [3]. Note that although the random MPQ-tickets produced by IRQ achieve good accuracy, the performance starts to degrade quickly when the average bitwidth is below 8-bit. We observe that MPQ-tickets consistently outperform other models in terms of best accuracies with low average bitwidth.

4.3. The hardware acceleration of MPQ-tickets

We choose FPGA chip XC7Z020 as the experiment platform which is a very common type currently to acceler-

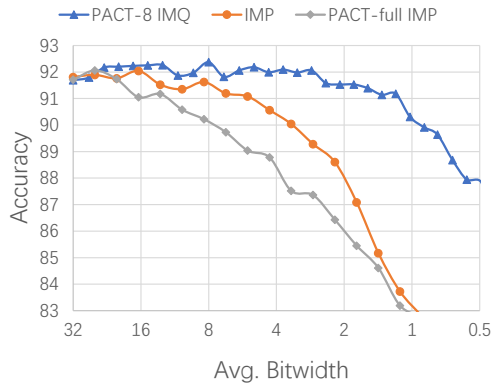


Figure 5. Comparison of IMP and PACT-full IMP. The performance degrades significantly in our experiment when activations are clipped even with learnable thresholds.

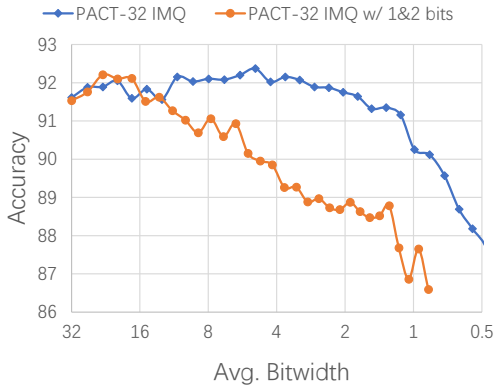


Figure 6. Comparison of IMQ when 1-bit and 2-bit are excluded from or included in quantization hierarchy. The performance degrades significantly when 1-bit and 2-bit are included.

ate neural network inference. The dominant frequency is 100MHz. We quantizes ResNet20s on CIFAR10 dataset utilizing the proposed MPQ-tickts. We explore the acceleration limits using three quantized activation (32-bit, 8-bit and 4-bit).

Figure 4 depicts the relationship of average bit-width and speed-up. The latency performance is normalized and compared with latency of full precision (32-bit). The trend can be seen from the Figure 4 that the lower the bit-width, the higher the speed-up. When the weight bit-width is lowest (2.29-bit), the latency performance is optimal, reaching 2.69×, 5.79× and 8.63× speed-up with 32bit, 8bit and 4bit activation, respectively.

The element-wise quantized operand improves the efficiency of processor, by making a single DSP slice calculates multiple MAC (Multiply-Accumulate) operations simultaneously. However, another benefit brought by the proposed MPQ-ticket is to reduce memory consumption, which addresses the bottleneck of data access. Since the element-wise quantization method compresses most of the weight into low-bit data, once data is accessed, more data can be fetched from and stored in the off-chip DDR. Figure 7 concludes that the memory consumption on FPGA reduces with the lower average bit-width. When the average bit-width is 2.29-bit, the memory consumption of ResNet20s on FPGA is only 0.07 MB, which outperforms 14.57× than the full precision model (1.02MB).

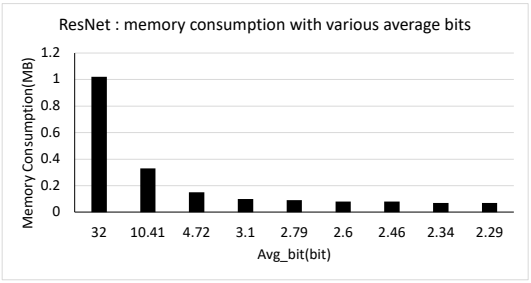


Figure 7. The memory consumption ResNet20s on FPGA with various average bit.

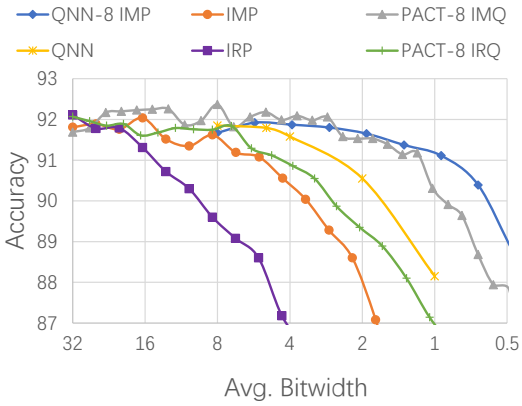


Figure 8. The curves of accuracy v.s. average bitwidth of different pruning or quantization methods. Note that *QNN-8 IMP* denotes IMP with the network initilized to an 8-bit QNN.

5. Conclusions

In this paper, we relax hard pruning of vanilla lottery tickets with soft pruning, and introduce IMQ which is soft version of IMP. With IMQ, we successfully validate the wide existence of MPQ-tickets, which outperform other strong baselines in terms of performance and compression trade-off. We further introduce customized hardware support to accelerate MPQ-tickets, and realize up to 8.x speed-up of inference speed, compared to the unaccelerable original lottery tickets due to the unstructured pruning nature. Future work may be directed to how to integrate 1-bit and 2-bit in the quantization hierarchy of IMQ for smoother soft pruning.

References

[1] Sung-En Chang, Yanyu Li, Mengshu Sun, Runbin Shi, Hayden K-H So, Xuehai Qian, Yanzhi Wang, and Xue Lin. Mix and match: A novel fpga-centric deep neural network quantization framework. In *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 208–220. IEEE, 2021.

[2] Weihan Chen, Peisong Wang, and Jian Cheng. Towards mixed-precision quantization of neural networks via constrained optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5350–5359, 2021.

[3] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018.

[4] Jack Choquette, Olivier Giroux, and Denis Foley. Volta: Performance and programmability. *IEEE Micro*, 38(2):42–52, 2018.

[5] Alberto Delmas, Patrick Judd, Sayeh Sharify, and Andreas Moshovos. Dynamic stripes: Exploiting the dynamic precision requirements of activation values in neural networks. *arXiv preprint arXiv:1706.00504*, 2017.

[6] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[7] Yu Gong, Zhihan Xu, Zhezhi He, Weifeng Zhang, Xiaobing Tu, Xiaoyao Liang, and Li Jiang. N3h-core: Neuron-designed neural network accelerator via fpga-based heterogeneous computing cores. In *Proceedings of the 2022 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pages 112–122, 2022.

[8] Patrick Judd, Jorge Albericio, Tayler Hetherington, Tor M Aamodt, and Andreas Moshovos. Stripes: Bit-serial deep neural network computing. In *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 1–12. IEEE, 2016.

[9] Qigong Sun, Licheng Jiao, Yan Ren, Xiufang Li, Fanhua Shang, and Fang Liu. Effective and fast: A novel sequential single path search for mixed-precision quantization. *arXiv preprint arXiv:2103.02904*, 2021.

[10] Frederick Tung and Greg Mori. Deep neural network compression by in-parallel pruning-quantization. *IEEE transactions on pattern analysis and machine intelligence*, 42(3):568–579, 2018.

[11] Yaman Umuroglu, Lahiru Rasnayake, and Magnus Själander. Bismo: A scalable bit-serial matrix multiplication overlay for reconfigurable computing. In *2018 28th International Conference on Field Programmable Logic and Applications (FPL)*, pages 307–3077. IEEE, 2018.

[12] Mart Van Baalen, Christos Louizos, Markus Nagel, Rana Ali Amjad, Ying Wang, Tijmen Blankevoort, and Max Welling. Bayesian bits: Unifying quantization and pruning. *Advances in neural information processing systems*, 33:5741–5752, 2020.

[13] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. Hardware-centric automl for mixed-precision quantization. *International Journal of Computer Vision*, 128(8):2035–2048, 2020.

[14] Ying Wang, Yadong Lu, and Tijmen Blankevoort. Differentiable joint pruning and quantization for hardware efficiency. In *European Conference on Computer Vision*, pages 259–277. Springer, 2020.

[15] Ziwei Wang, Han Xiao, Jiwen Lu, and Jie Zhou. Generalizable mixed-precision quantization via attribution rank preservation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5291–5300, 2021.

[16] Bichen Wu, Yanghan Wang, Peizhao Zhang, Yuandong Tian, Peter Vajda, and Kurt Keutzer. Mixed precision quantization of convnets via differentiable neural architecture search. *arXiv preprint arXiv:1812.00090*, 2018.

[17] Huanrui Yang, Lin Duan, Yiran Chen, and Hai Li. Bsq: Exploring bit-level sparsity for mixed-precision neural network quantization. *arXiv preprint arXiv:2102.10462*, 2021.

[18] Haibao Yu, Qi Han, Jianbo Li, Jianping Shi, Guangliang Cheng, and Bin Fan. Search what you want: Barrier panelty nas for mixed precision quantization. In *European Conference on Computer Vision*, pages 1–16. Springer, 2020.

[19] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016.